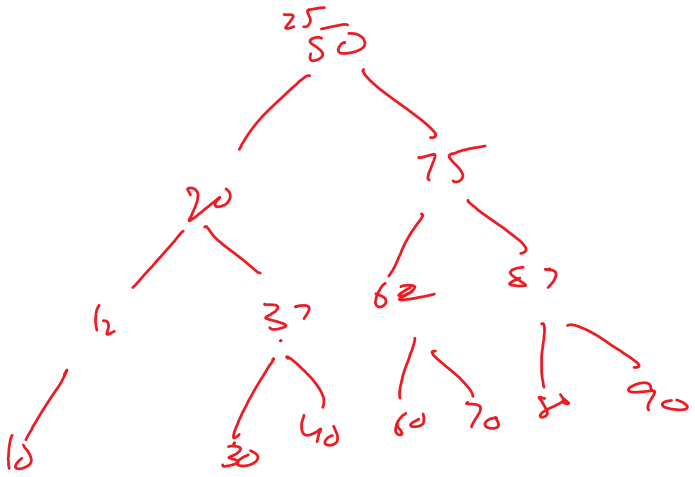
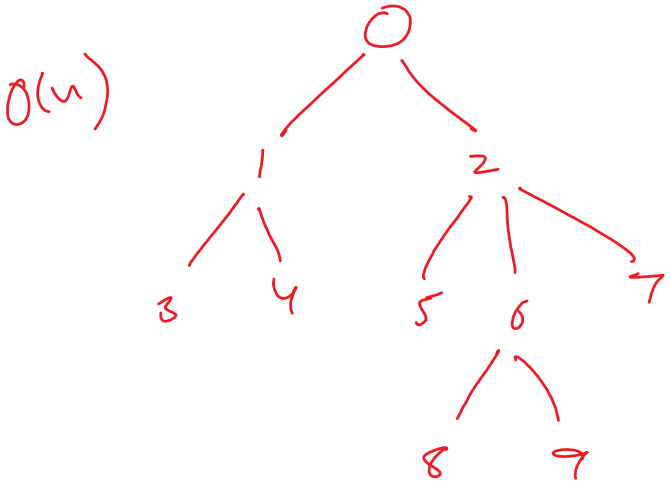
Wednesday, June 1, 2022 9:17 PM




```
public TreeNode deleteNode(TreeNode root, int key) {
    if(key > root.val){
        root.right = deleteNode(root.right, key);
        return root;
    } else if(key < root.val){
        root.left = deleteNode(root.left, key);
        return root;
    } else {
        if(root.left != null && root.right != null){

        } else if(root.left != null){
            return root.left;
        } else if(root.right != null){
            return root.right;
        } else {
            return null;
        }
    }
}
```



1 + 1+2 +2 +2+2+2 +3+3

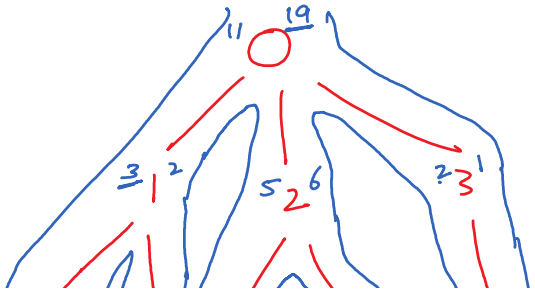
1.3+1.2+3.3 +2.4

3 + 2 + 9 + 8

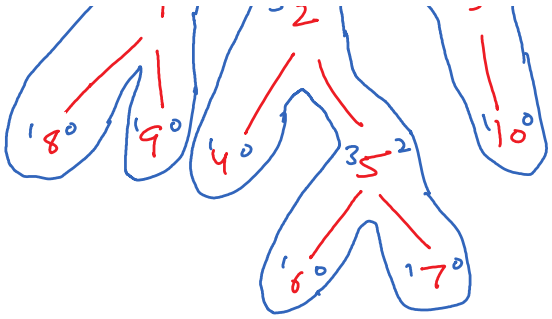
1+2+2+3+3+3+4+5+5

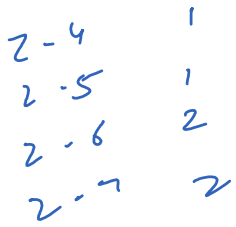
9:38 - 9:48

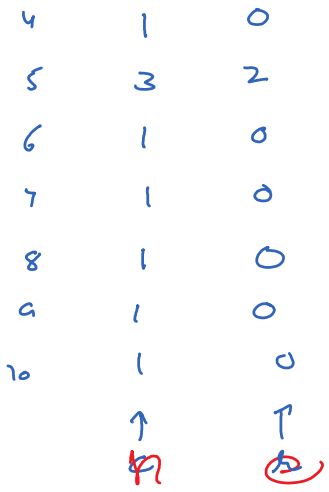
[18/0 , 22/1 , -/2 , -/3 , -/4 , -/5 , -/6 , -/7 , -/8 , 28/9]

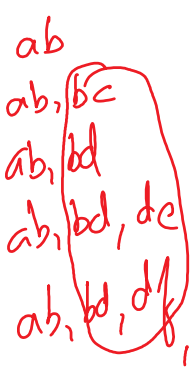


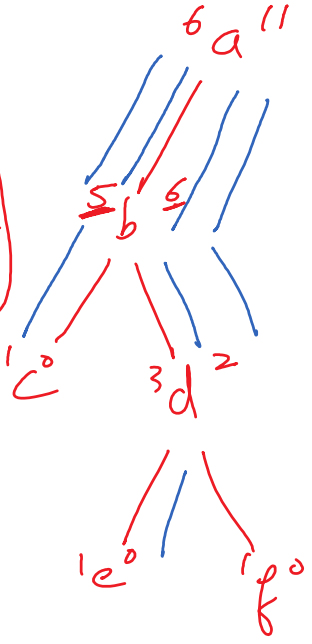
0	11	19
1	3	2
2	5	6
3	2	1
4	1	0
5	3	2

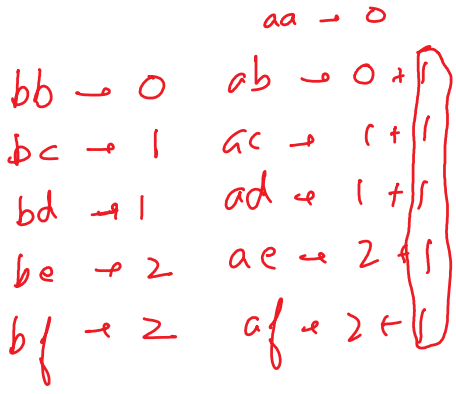


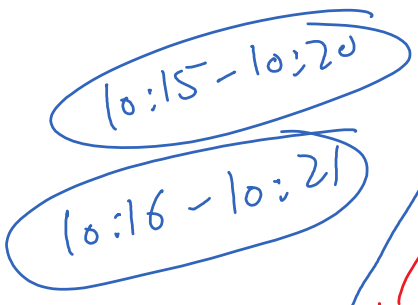


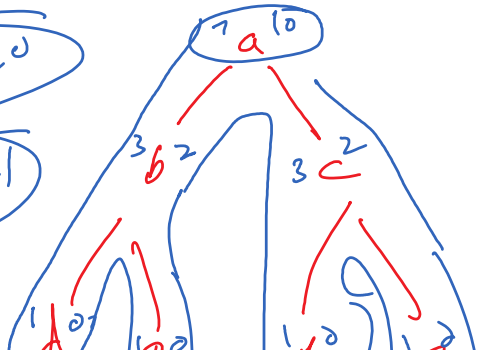


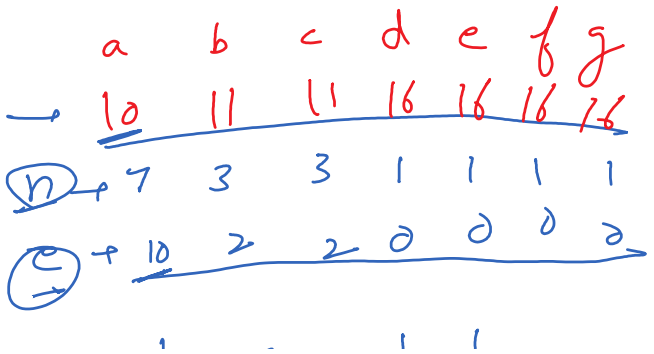


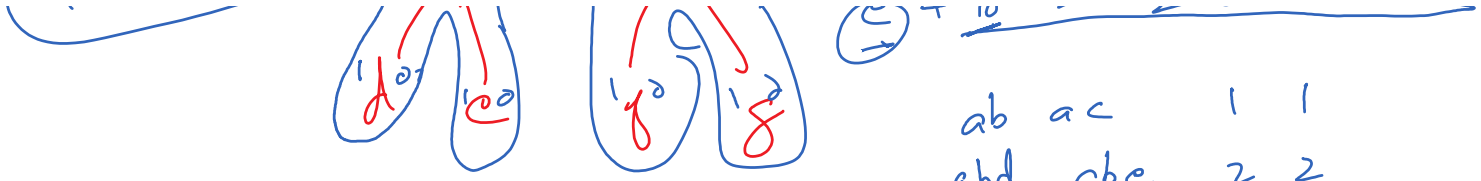












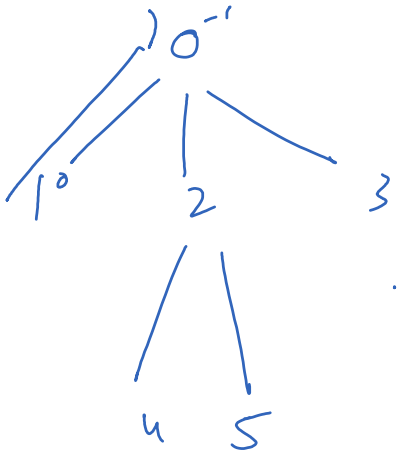
bd, be → ab, abd, abe

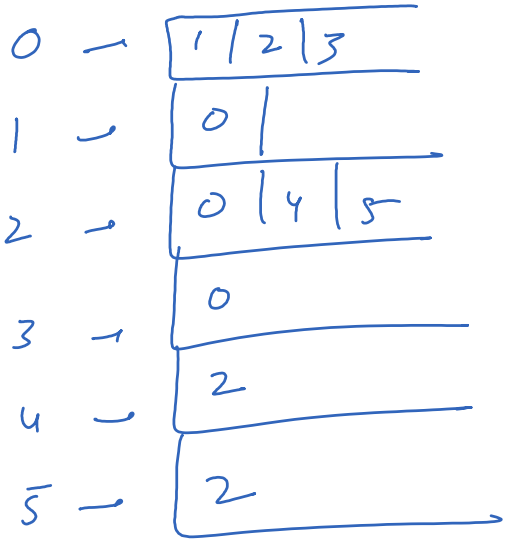
cf, cg

ab ac 1 1

abd abe 2 2

acf acg 2 2

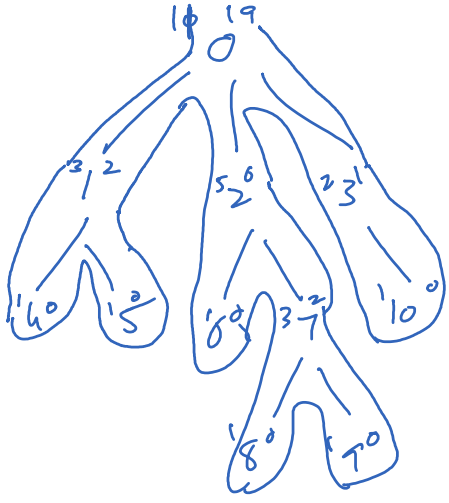




```
    HashSet<Integer>[] graph = new HashSet[n];
    for(int i = 0; i < graph.length; i++){
        graph[i] = new HashSet<>();
    }

    for(int i = 0; i < edges.length; i++){
        int u = edges[i][0];
        int v = edges[i][1];

        graph[u].add(v);
        graph[v].add(u);
    }

    int[] nodes = new int[n];
    int[] res = new int[n];
    helper1(graph, nodes, res, 0, -1);
}

public void helper1(HashSet<Integer>[] graph, int[] nodes, int[] res, int src, int par){
    for(int nbr: graph[src]){
        if(nbr != par){
            helper1(graph, nodes, res, nbr, src);
```

0 → 1,2,3
1 → 0,4,5
2 → 0,6,7
3 → 0,10
4 → 1
5 → 1
6 → 2
7 → 2,8,9
8 → 7
9 → 7
10 → 3

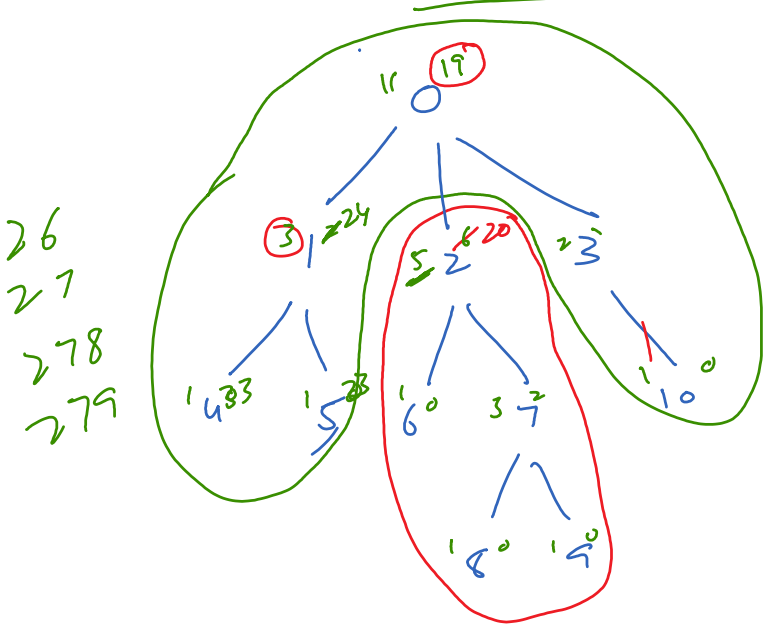
n 19 19
1 0
1 0

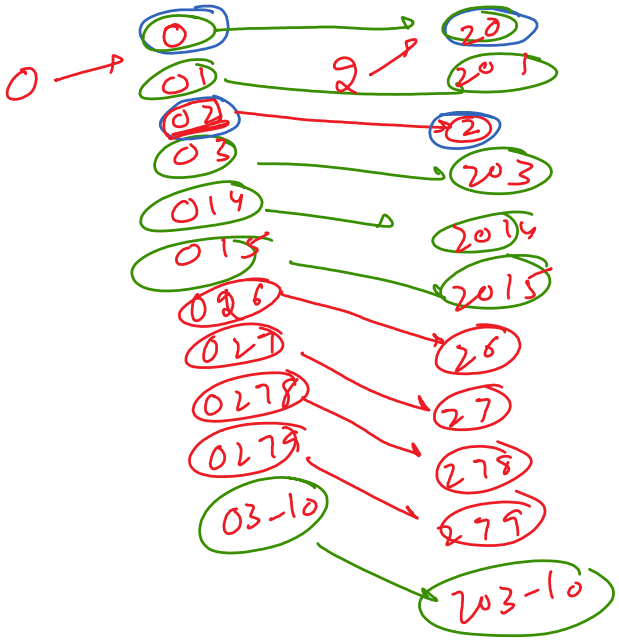


```java
public void helper1(HashSet<Integer>[] graph, int[] nodes, int[] res, int src, int par){
    for(int nbr: graph[src]){
        if(nbr != par){
            helper1(graph, nodes, res, nbr, src);
            nodes[src] += nodes[nbr];
            res[src] += nodes[nbr] + res[nbr];
        }
    }

    nodes[src]++;
}
```

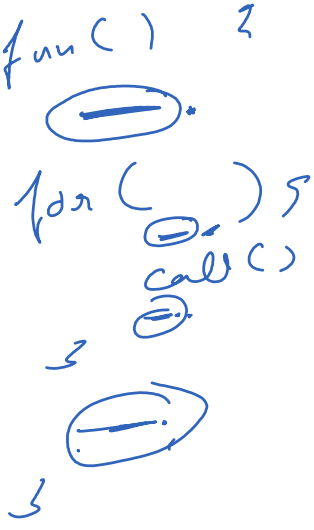
10:42 - 10:47

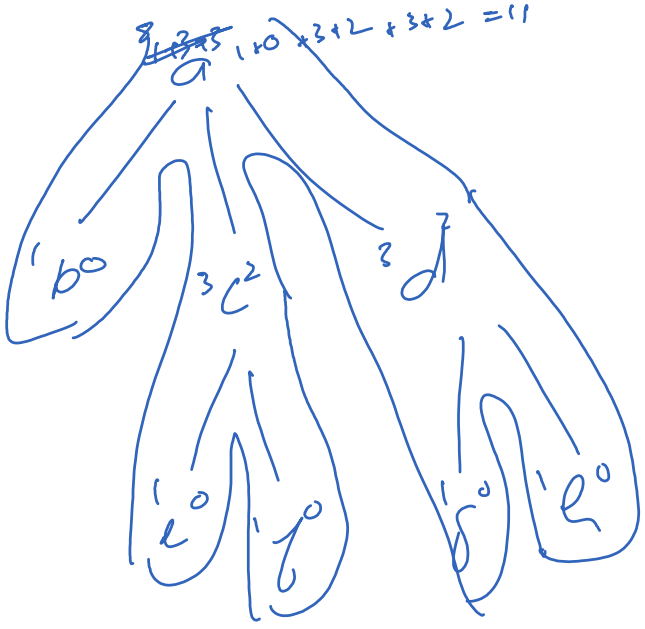
n.l - n[4]

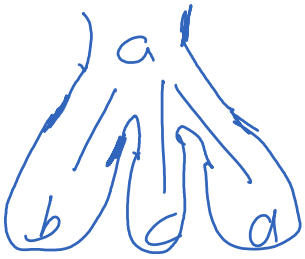




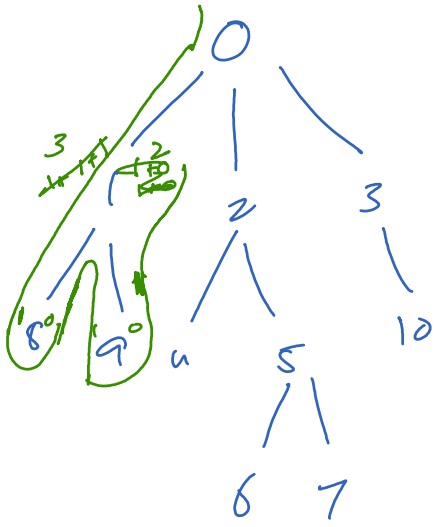
19 → 5 + 6





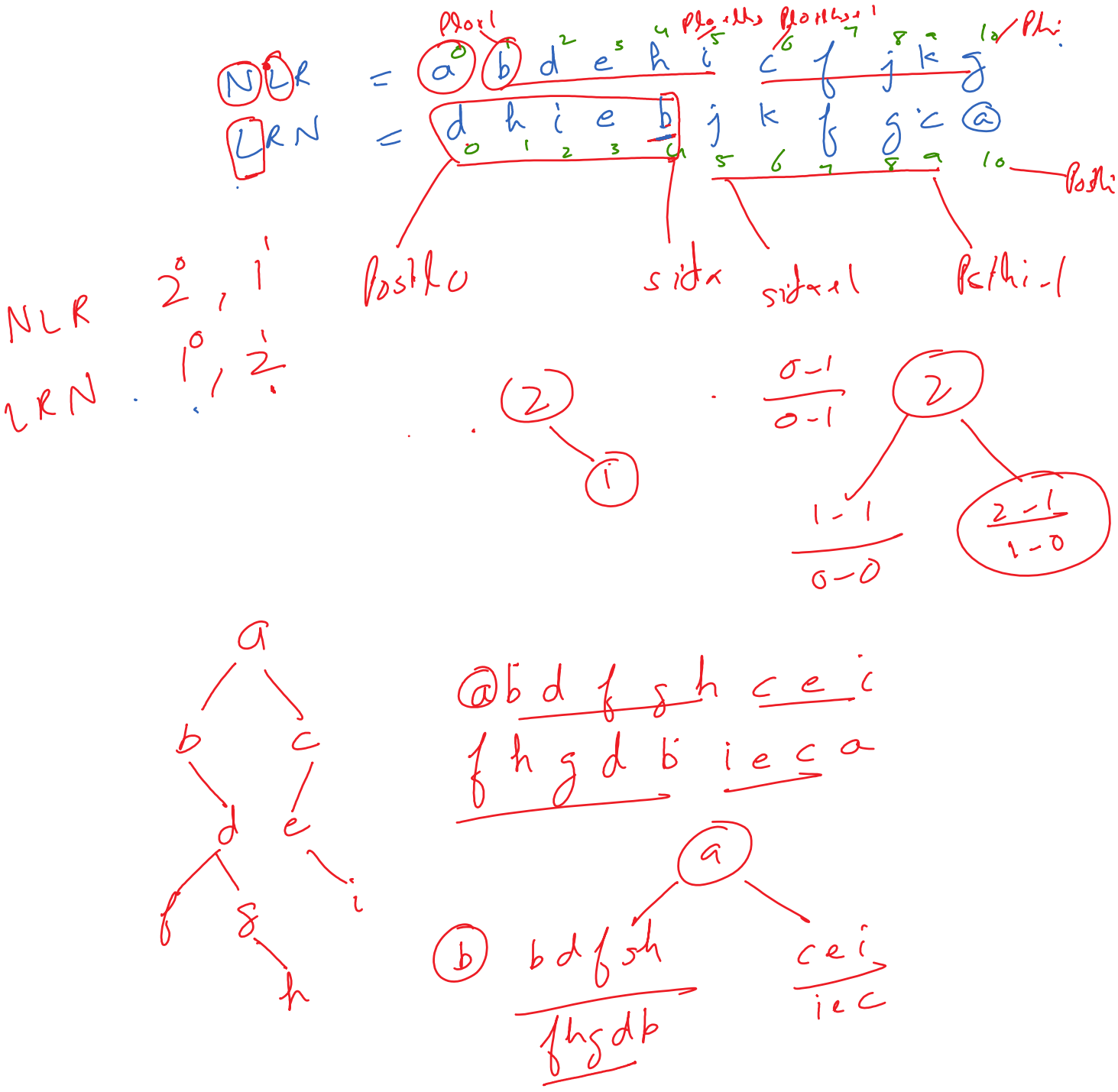


```java
public void helper1(HashSet<Integer>[] graph, int[] nodes,
    for(int nbr: graph[src]){
        if(nbr != par){
            helper1(graph, nodes, res, nbr, src);
            nodes[src] += nodes[nbr];
            res[src] += nodes[nbr] + res[nbr];
        }
    }

    nodes[src]++;
}
```

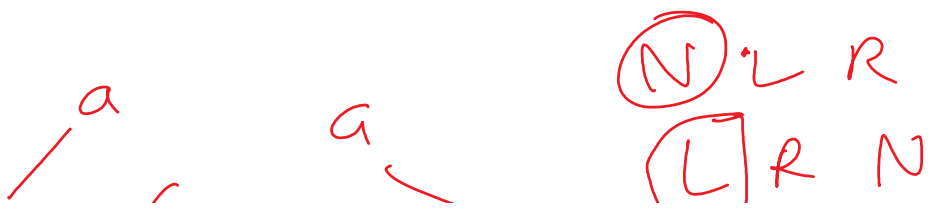


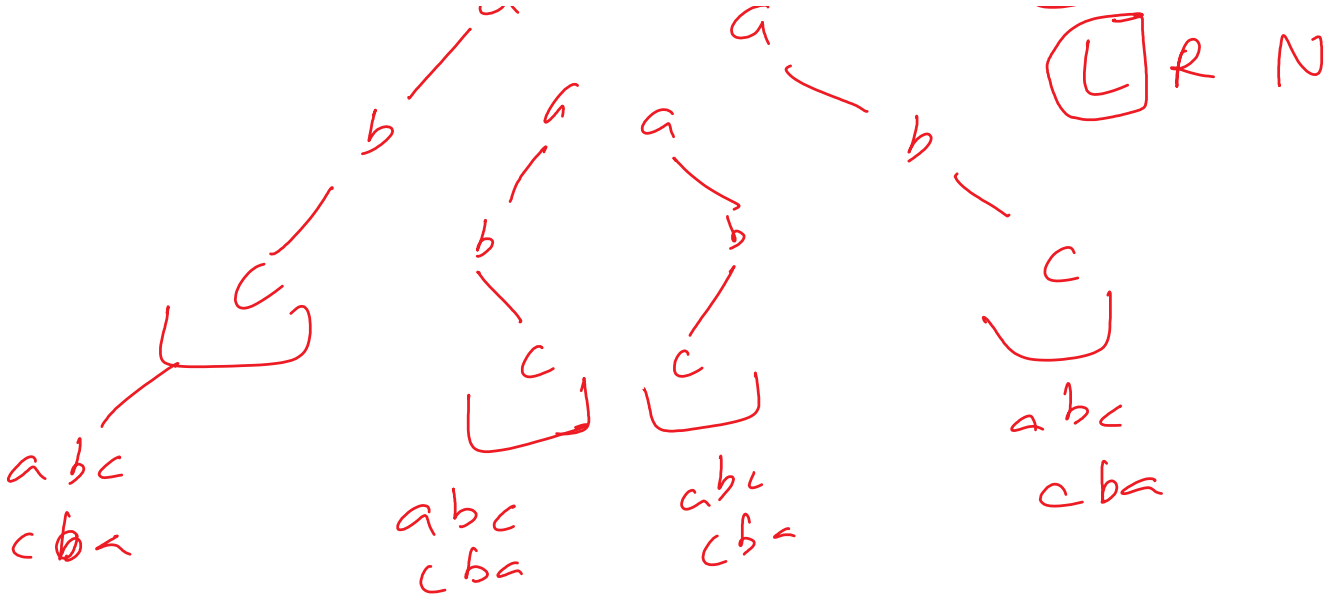

```java
public void helper2(ArrayList<Integer>[] graph, int[] nodes, int[] res, int src, int par){
    for(int nbr: graph[src]){
        if(nbr != par){
            res[nbr] = res[src] + (nodes.length - nodes[nbr]) - (nodes[nbr]);
            helper2(graph, nodes, res, nbr, src);
        }
    }
}
```

```java
public void helper1(ArrayList<Integer>[] graph, int[] nodes, int[] res, int src, int par){
    for(int nbr: graph[src]){
        if(nbr != par){
            helper1(graph, nodes, res, nbr, src);
            nodes[src] += nodes[nbr];
            res[src] += nodes[nbr] + res[nbr];
        }
    }

    nodes[src]++;
}
```


NLR = a b d e h i c f j k g
LRN = d h i e b j k f g c a
Postlo
sidx
sidx+1
Pothi-1
NLR 2° , 1'
LRN 1° , 2'
2
1
0-1
0-1
2
1-1
0-0
2-1
1-0

a
b c
d e
f g i
h
@ b d f g h c e i
f h g d b i e c a
a
b
b d f g h
f h g d b
c e i
i e c


a
a
N L R
L R N


L
R
N
abc
cba
abc
cba
abc
cba
abc
cba